Backpropagation Training

Michael J. Watts

http://mike.watts.net.nz

Lecture Outline

- Backpropagation training
- Error calculation
- Error surface
- Pattern vs. Batch mode training
- Restrictions of backprop
- Problems with backprop

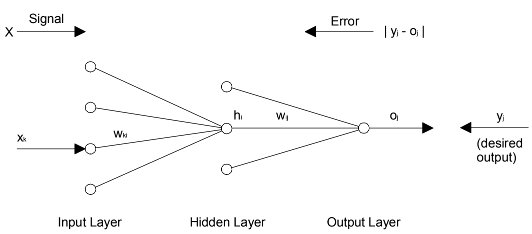
Backpropagation Training

- Backpropagation of error training
- Also known as backprop or BP
- A supervised learning algorithm
- Outputs of network are compared to desired outputs

Backpropagation Training

- Error calculated
- Error propagated backwards over the network
- Error used to calculate weight changes

Backpropagation Training



Backpropagation Training

Forward pass:

BF1. Apply an input vector **x** and its corresponding output vector **y** (the desired output).

BF2. Propagate forward the input signals through all the neurons in all the layers and calculate the output signals.

BF3. Calculate the Err_j for every output neuron j as for example:
$Err_j = y_j - o_j$, where y_j is the jth element of the desired output vector **y**.

Backward pass:

BB1. Adjust the weights between the intermediate neurons i and output neurons j according to the calculated error:
$\Delta w_{ij}(t+1) = lrate.\ o_j(1 - o_j).\ Err_j.\ o_i + momentum.\ \Delta w_{ij}(t)$

BB2. Calculate the error Err_i for neurons i in the intermediate layer:
$Err_i = \quad Err_j.\ w_{ij}$

BB3. Propagate the error back to the neurons k of lower level:
$\Delta w_{ki}(t+1) = lrate.o_i(1 - o_i).\ Err_i.x_k + momentum.\ \Delta w_{ki}(t)$

Backpropagation Training

$$w_{i,j}(t+1) = w_{i,j}(t) + \eta\Delta w_{i,j} + \alpha\Delta w_{i,j}(t)$$

- where:
  - delta is the weight change
  - eta is the learning rate
  - alpha is the momentum term

Backpropagation Training

$$\Delta w_{i,j}(t+1) = -\frac{\partial E}{\partial w_{i,j}(t)}$$

- Where

$\Delta w_{i.i}$ • Is the change to the weight

$-\frac{\partial E}{\partial w_{i,j}(t)}$ is the partial derivative of the error *E* with Respect to the weight *w*

Error Calculation

- Several different error measures exist
  - applied over examples or complete training set
- Sum Squared Error
  - SSE
- Mean Squared Error
  - MSE

Error Calculation

- Sum Squared Error
  - SSE
  - Measured over entire data set

$$E = \frac{1}{2}\sum_p \sum_i (d_{pi} - a_{pi})^2$$

Error Calculation

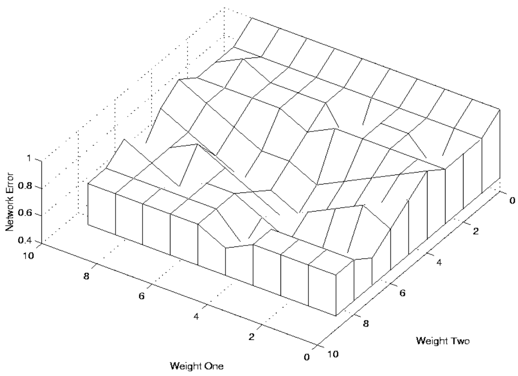
- Mean Squared Error
  - measured over individual examples
  - reduces errors over multiple outputs to a single value

$$Err_j(p) = \frac{(d_j^{(p)} - a_j^{(p)})^2}{2}$$

Error Surface

- Plot of network error against weight values
- Consider network as a function that returns an error
- Each connection weight is one parameter of this function
- Low points in surface are local minima
- Lowest is the global minimum

Error Surface



Error Surface

- At any time *t* the network is at one point on the error surface
- Movement across surface from time *t* to *t*+1 is because of BP rule
- Network moves “downhill” to points of lower error
- BP rule is like gravity

Error Surface

- Learning rate is like a multiplier on gravity
- Determines how fast the network will move downhill
- Network can get stuck in a dip
  - stuck in a local minimum
  - low local error, but not lowest global error

Error Surface

- Too low learning rate = slow learning
- Too high = high chance of getting stuck

Error Surface

- Momentum is like the momentum of a mass
- Once in motion, it will keep moving
- Prevents sudden changes in direction
- Momentum can carry the network out of a local minimum

Error Surface

- Not enough momentum = less chance of escaping a local minimum
- Too much momentum means network can fly out of global minimum

Pattern vs Batch Training

- Also known as
  - batch and online
  - offline and online
- Pattern mode applies weight deltas after each example
- Batch accumulates deltas and applies them all at once

Pattern vs Batch Training

- Batch mode is closer to true gradient descent
  - requires smaller learning rate
- Step size is smaller
- Smooth traversal of the error surface
- Requires many epochs

Pattern vs Batch Training

- Pattern mode is easier to implement
- Requires shuffling of training set
- Not simple gradient descent
  - Might not take a direct downward path
- Requires a small step size (learning rate) to avoid getting stuck

Restrictions on Backprop

- Error and activation functions must be differentiable
- Hard threshold functions cannot be used
  - e.g. step (Heaviside) function
- Cannot model discontinuous functions
- Mixed activation functions cannot be used

Problems with Backprop

- Time consuming
  - hundreds or thousands of epochs may be required
  - variants of BP address this
    - quickprop
- Selection of parameters is difficult
  - epochs, learning rate and momentum

Problems with Backprop

- Local minima
  - can easily get stuck in a locally minimal error
- Overfitting
  - can overlearn the training data
  - lose the ability to generalise
- Sensitive to the MLP topology
  - number of connections
    - “free parameters”

Summary

- Backprop is a supervised learning algorithm
- Gradient descent reduction of errors
- Error surface is a multidimensional surface that describes the performance of the network in relation to the weight

Summary

- Traversal of the error surface is influenced by the learning rate and momentum parameters
- Pattern vs. Batch mode training
  - difference is when deltas are applied
- Backprop has some problems
  - local minima
  - Overfitting